

Demo: 6TiSCH on SC

μ

M, Running a Synchronized Protocol Stack without Crystals

Tengfei Chang, Thomas Watteyne, Brad Wheeler, Filip Maksimovic, Osama Khan, Sahar Mesri, Lydia Lee, Kris Pister, Ioana Suci, Xavier Vilajosana

► To cite this version:

Tengfei Chang, Thomas Watteyne, Brad Wheeler, Filip Maksimovic, Osama Khan, et al.. Demo: 6TiSCH on SC

μ

M, Running a Synchronized Protocol Stack without Crystals. EWSN 2020 - ACM International Conference on Embedded Wireless Systems and Networks, Feb 2020, Lyon, France. hal-02436262

HAL Id: hal-02436262

<https://hal.inria.fr/hal-02436262>

Submitted on 12 Jan 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Demo: 6TiSCH on SC_μM, Running a Synchronized Protocol Stack without Crystals

Tengfei Chang, Thomas Watteyne
Inria, France

{*tengfei.chang, thomas.watteyne*}@inria.fr

Brad Wheeler, Filip Maksimovic, Osama Khan, Sahar Mesri, Lydia Lee, Kris Pister
UC Berkeley, CA, USA

{*brad.wheeler, fil, oukhan, smesri, lydia.lee, ksjp*}@berkeley.edu

Ioana Suci, Xavier Vilajosana
Univ. Oberta de Catalunya, Spain

{*isuciu0, xvilajosana*}@uoc.edu

Abstract

We report the first time-synchronized protocol stack running on a crystal-free device. We use an early prototype of the Single-Chip micro Mote, SC_μM, a single-chip $2 \times 3 \text{ mm}^2$ mote-on-a-chip, which features an ARM Cortex-M0 micro-controller and an IEEE802.15.4 radio. This prototype consists of a FPGA version of the micro-controller, connected to the SC_μM chip which implements the radio front-end. We port OpenWSN, a reference implementation of a synchronized protocol stack, onto SC_μM. we first calibrate the oscillators by receiving packets via the on-chip optical receiver and RF transceiver so that SC_μM can send frames to an off-the-shelf IEEE802.15.4 radio. We then use a digital trimming compensation algorithm based on tick skipping to compensate the frequency converting rounding error. This allows us to run a full-featured standards-compliant 6TiSCH network between one SC_μM and one OpenMote, a firm step toward realizing the smart dust vision of ultra-small and cheap ubiquitous wireless devices.

Keywords

Crystal-free, 6TiSCH, SC_μM, smart dust.

1 6TiSCH and SC_μM

Time Synchronized Channel Hopping (TSCH) is at the core of all main industrial standards, including WirelessHART, ISA100.11a and IEEE802.15.4. 6TiSCH [3] protocol stack is the latest such standardization efforts, lead by the Internet Engineering Task Force (IETF). It combines the industrial performance of IEEE802.15.4 TSCH, with the

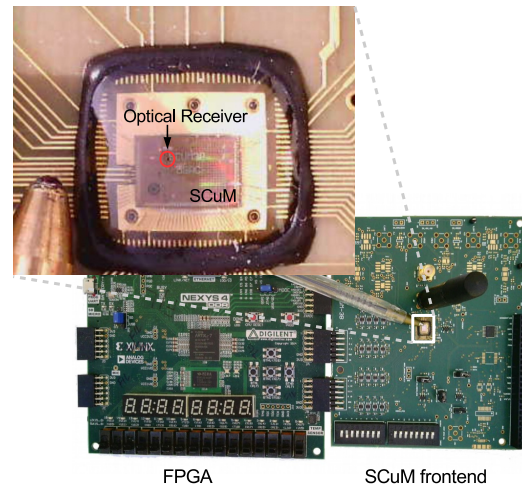


Figure 1. The Single Chip Micro-Mote (SC_μM) is a $2 \times 3 \text{ mm}^2$ mote-on-a-chip. It features an ARM Cortex-M0 micro-controller, an IEEE802.15.4 radio, and an optical bootloader. While SC_μM runs with no external components, it is shown here on its development board. In this setup, we use an FPGA board to implement the digital part (including the Cortex-M0 micro-controller), and use the analog front-end of the SC_μM chip.

IETF upper stack for IoT devices. As depicted in Fig. 2, this upper stack includes CoAP, UDP, RPL and 6LoWPAN.

The commercial IEEE802.15.4-compliant chips running these standards use stable oscillators as a time reference. Typical those crystal oscillators drift rates, i.e. the inaccuracy of the frequency, is in the 10-30 ppm (parts-per-million) range. The problem of needing a crystal is cost and space. While the crystal itself might be relatively cheap (in the USD 0.50 range), using them requires one to make a printed circuit board to assemble the crystal to the chip, which consumes space and increases cost.

The Single Chip micro-Mote, or SC_μM, is a crystal-free

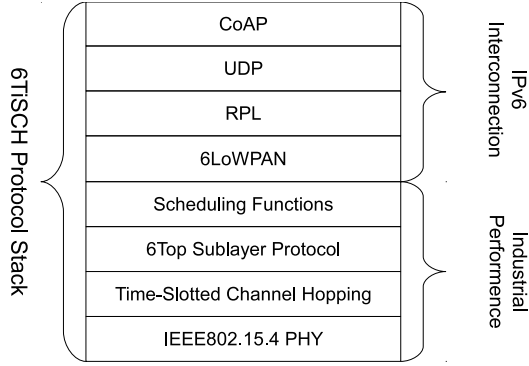


Figure 2. The 6TiSCH stack. The upper stack provides IPv6 connectivity. The lower stack, through TSCH, provides industrial-level performance.

chip we taped out in 2019. It is a 2×3 mm² single-chip crystal-free mote-on-chip which contains an ARM Cortex-M0 micro-controller, a 2.4 GHz IEEE802.15.4 radio, and an optical receiver for optical programming. Fig. 1 shows SC_μM on the board we use to develop/debug it. The digital part of SC_μM is implemented over a FPGA board (indicated on the left of the figure) and the analog part is implemented over the chip. Loading code into the chip is done optically by using an external board which blinks an LED close to the optical receiver on SC_μM.

The goal of this paper is to show a 6TiSCH network composed of one SC_μM and one OpenMote [2]. The challenge is that SC_μM does not have an accurate sense of time, and therefore derives its time reference from OpenMote. The following section describes how SC_μM tunes the frequency it communicates on, and how we calibrate its clocks.

2 Frequency Synthesis and Clock Calibration

First we need to give SC_μM a rough time reference so it can send frames that OpenMote can receive. The frequency of each of the oscillators is tunable. We designed the code running on the optical programmer board in such a way that, at the end of the bootloading process, the optical programmer repeatedly sends a sequence that causes an `OPTICAL_ISR` interrupt to be generated on SC_μM. This interrupt fires every 100 ms for 2.5 s. While this is happening, on SC_μM, all the clocks are running. By recording the counter value of each of the clocks, and knowing the interval between interrupts, SC_μM calibrates each of the oscillators.

Following this coarse calibration using the optical programmer, SC_μM can also calibrate against the OpenMote. We do this offline, i.e. this calibration is done once, the result of which is reused the next time SC_μM is programmed. For this calibration, SC_μM sends frames on channel 11 (2.405 GHz) to OpenMote. OpenMote is programmed to listen on that channel, and prints over its serial port the value of its `XREG_FREQEST` register, which indicates the frequency offset of the incoming signal. According to that value, we manually tune the 2.4GHz oscillator of SC_μM, to minimize that frequency offset. This procedure is repeated for each SC_μM board, as each has slightly different tuning parameters. This is illustrated in Fig. 3.

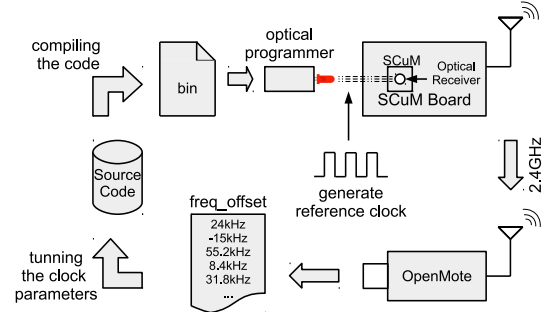


Figure 3. Setup for tuning the communication frequency of SC_μM. SC_μM transmits frames to OpenMote, which logs the frequency offset for each frame it receives. These offsets are then used to manually tune the 2.4GHz oscillator of SC_μM, which is used to select the transmit frequency, to minimize the mean frequency offset.

While SC_μM is running the 6TiSCH stack, it keeps synchronized to the OpenMote. Part of that is making sure the boundaries of its TSCH slots are aligned in time with that of OpenMote. The frequency of the timer used for implementing TSCH slot state machine runs at 32 kHz. However, RFTimer, the timer SC_μM used for the same purpose, runs at 500 kHz. This means the OpenWSN port to SC_μM divides down the 500 kHz RFTimer so it appears as a 32 kHz clock source to the otherwise unmodified OpenWSN stack implementation. Since $500/32 = 15.625$, the integer division applied in the port results in a rounding error. This means the slot length on SC_μM is slightly different than the slot length of OpenMote. As is, this difference in slot length results in an apparent relative drift between OpenMote and SC_μM. We therefore implement a digital trimming (tick skipping) compensation algorithm, explained in [1].

3 Conclusion

This paper provides the first example of a synchronized network protocol (6TiSCH) running on a crystal-free device, the Single Chip micro-Mote (SC_μM), a 2×3 mm² crystal-free mote-on-a-chip. SC_μM features an ARM Cortex-M0 micro-controller and an IEEE802.15.4 radio. It first listens to a blinking LED to provide coarse calibration of its oscillators. Then using an OpenMote, which report the frequency offset, provides a second level of more precise tuning. Finally, as SC_μM and OpenMote are communicating, the OpenWSN port on SC_μM uses a digital trimming compensation algorithm based on tick skipping to counteract a rounding error caused by frequency converting. This allows a synchronized fully functional 6TiSCH network to form between SC_μM and OpenMote.

4 References

- [1] T. Chang, T. Watteyne, K. Pister, and Q. Wang. Adaptive synchronization in multi-hop TSCH networks. *Computer Networks*, pages 165–176, 15 January 2015.
- [2] X. Vilajosana, P. Tuset, T. Watteyne, and K. Pister. OpenMote: Open-Source Prototyping Platform for the Industrial IoT. In *International Conference on Ad Hoc Networks*, San Remo, Italy, 2015. Springer.
- [3] X. Vilajosana, T. Watteyne, M. Vučinić, T. Chang, and K. Pister. 6TiSCH: Industrial Performance for IPv6 Internet-of-Things Networks. *Proceedings of the IEEE*, pages 1–13, 11 April 2019.